REMARKS

Claims 1-26 are pending in the present application. Reconsideration of the claims is respectfully requested.

I. 35 U.S.C. § 103, Alleged Obviousness, Claims 1-26

The Office Action rejects claims 1-26 under 35 U.S.C. § 103(a) as being unpatentable over Chatwani et al. (U.S. Patent No. 5,729,685) in view of Wiley et al. (U.S. Patent No. 5,687,320). This rejection is respectfully traversed.

As to claims 1, 14, and 25, the Office Action states:

> As per claims 1, 14, and 25, Chatwani discloses a method, apparatus and computer program product for retrieving client boot information in a network environment with multiple boot servers (col 4, lines 15-18), comprising:
>
> initiating at a client an initial request (col 29,lines 36-44) for client configuration information (alternative embodiment, col 29, lines 55-67);
>
> sending from the client the initial request for client configuration information to a first boot server (CMS processor is client, fig 26, col 23 lines 17-26, col 34, lines 30-57 and col 29, lines 55-67);
>
> receiving at the client a boot server list (CMS is acting as a client, col 29) if the client configuration information is not found on the first boot server (CMS functionality is to determine correct boot code, boot server, optical path and perform load balancing, col 30, lines 61-67 and col 32, lines1-11); and
>
> sending from the client a configuration information request (col 26, lines 12-16) for the client configuration (col 26, lines 12-16) information to each server (col 12, lines 5-6) in the boot server list (col 33, lines 16-54, first to next shows the order) until the client configuration information is found (col 32, lines 65-67) or a request has been sent to every server in the boot server list (fig 23(a)-24, clearly shows the CMS is identifying a boot server based on the BFQ message, col 33, lines 33-45,col 33, lines 16-54). Chatwani may not be using the same terms as claimed, such as initiating at a client an initial request for client configuration information; the client information is not found on the first boot server, and sending a boot server list to the client if the information is not found. However, initiating at a client an initial request for client configuration information (it is a part of initialization, in a client-server model client initiates the request and serves the request), if the client information is not found on the first boot server, and sending a boot server list to the client if the information is not found

> (error handling if very well known in the art, this may be default choice) are very well known in the art. Wiley, for example, discloses as initiating at a client an initial request for client configuration information (col 4, lines 51-60); the client information is not found on the first boot server (col 4, lines 6-24), and sending a boot server list to the client if the information is not found (col 4, lines 6-24). It would have been obvious to one of ordinary skill in the art at the time of the invention was made to combine the teachings of Chatwani and Wiley. The motivation would have been to have system where client can obtain an alternate boot sever by requesting the boot server list in the case of primary boot server is affected.

Office Action dated January 6, 2005, pages 3-5.

Claim 1, which is representative of the other rejected independent claims 14 and 25 with regard to similarly recited subject matter, reads as follows:

> 1. A method for retrieving client boot information in a network environment with multiple boot servers, comprising:
>
> initiating at a client an initial request for client configuration information;
>
> sending from the client the initial request for client configuration information to a first boot server;
>
> receiving at the client a boot server list if the client configuration information is not found on the first boot server; and
>
> sending from the client a configuration information request for the client configuration information to each server in the boot server list until the client configuration information is found or a request has been sent to every server in the boot server list.

Chatwani and Wiley, taken alone or in combination, fail to teach or suggest if the client configuration information is not found on the first boot server, sending from the client a list request for a boot server list to the first boot server, receiving at the client the boot server list, and sending from the client a configuration information request for the client configuration information to each server in the boot server list until the client configuration information is found or a request has been sent to every server in the boot server list.

Chatwani is directed to a method for automatically determining the topology of the network. Chatwani provides for each switch in the network, transmitting on each of its ports, link advertisement messages. The link advertisement messages are received by neighbor switches and forwarded to a topology manager. The topology manager

constructs network topology profile information based on received link advertisement messages. Further, the topology manager is able to verify bi-direction links based on the received link advertisement messages.

The Office Action alleges that Chatwani teaches initiating at a client an initial request for client configuration information and sending from a client an initial request for client configuration information to a first boot server at column 29, lines 36-44, column 29, lines 55-67, Figure 26, column 23 lines 17-26, and column 34, lines 30-57, which read as follows:

> A. Overview of the bootstrapping process
>
> FIG. 20 is useful for providing an overall flow diagram of the method of boostrapping utilized in the described system.
>
> When a switch is first powered up (or when its power is restored after, for example, a power failure or when it is reconnected to the network after being disconnected for some reason), the switch begins sending out what will be termed path access query (PAQ) messages, block 2001. The format for the PAQ messages, as transmitted by the booting switch, is found with reference to FIG. 21(a). PAQ messages are transmitted over each of booting switches ports over the meta-topology channel and is received by neighboring switches (assuming there are any operational neighboring swiches). The PAQ message is transmitted over the topology service channel of each of the neighbor switch's VSPs through the master switch to the CMS. The format for the messages transmitted by the neighboring switches is given by FIG. 21(b). The format for messages transmitted from the master switch to the CMS is given by FIG. 21(c).
>
> In response to receiving each PAQ message, the CMS transmits a path access response (PAR) message onto the topology service channel of the neighbor switch. The format for the PAR messages as transmitted by the CMS is given by FIG. 21(d). This message is received by the neighbor switch and transmitted on the meta-topology channel to the booting switch using the format given by FIG. 21(d), block 2002.
>
> It is worthwhile noting that, in alternative embodiments, some device other than the CMS may function to coordinate providing boot code to the booting switch. For example, a boot server may, in some embodiments, be implemented as an ATM device capable of communicating directly onto the ATM network. In such a case, the boot server may respond to the PAQ messages directly by transmitting PAR messages in response to receiving PAQ messages in a manner similar to what is described for the CMS. Therefore, the functions described herein for the CMS in relation to managing a boot code transfer may sometimes be referred to as a boot management. Further, the CMS when performing

boot functions (and variations of the CMS such as a boot server directly providing boot management) may be referred to as a boot manager.

(Chatwani, column 29, line 28 to column 30, line 2)

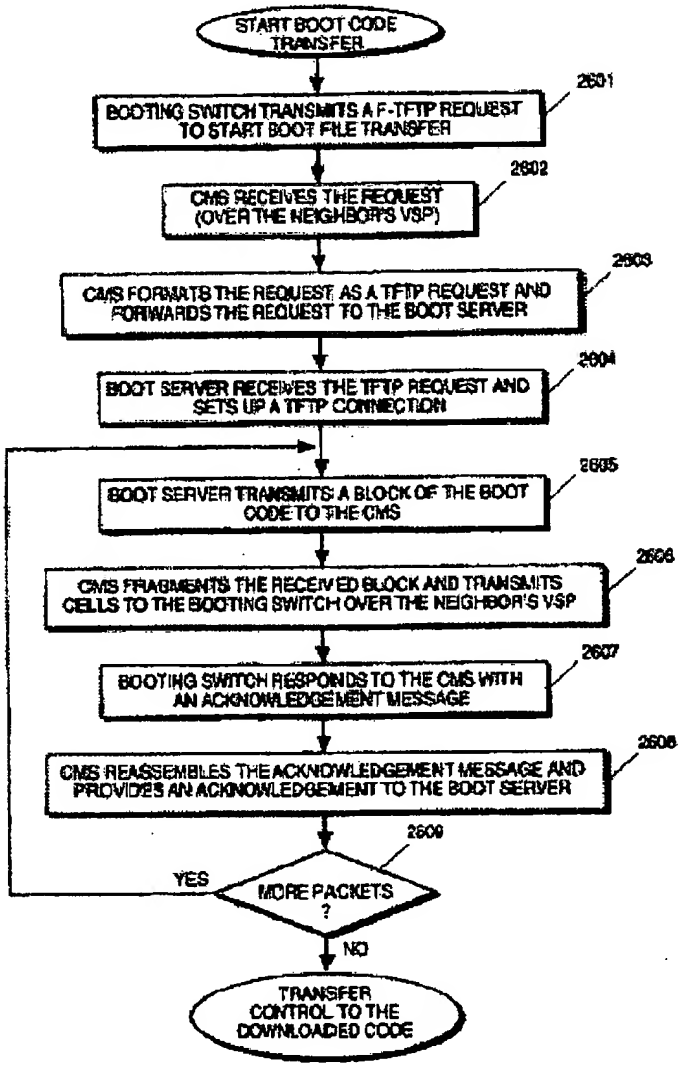

Figure 26

(Chatwani, Figure 26)

The described embodiment utilizes the above-described VSP concept to allow for dynamic registration of clients and to allow for client-to-client communication through cloud 1301. In broad terms, the dynamic client discovery process of the described embodiment starts by a client

> attempting to register itself when it is attached to the network and, in response to these attempts, the CMS registering the client and allowing later identification of the client by some address type (which may be either a logical or physical address) or by identification of a switch, module and port.

(Chatwani, Column 23, lines 17-26)

> FIG. 26 is useful for providing a more complete description of the process for transfer of the boot file. The booting switch initially generates a request for transmission of its boot file using the F-TFTP format, block 2601. The CMS receives the request (over the neighbor's VSP as has been described), block 2602, and the CMS formats a TFTP request and forwards the request to the boot server, block 2603. The boot server receives the TFTP request and sets up the TFTP connection, block 2604. The boot server then transmits a block of boot code to the CMS, block 2605. The CMS is responsible for fragmenting the received block and transmitting cells containing information from the received block to the booting switch over the neighbor's VSP, block 2606. The booting switch responds to the CMS with an acknowledgement message when all cells comprising the received block have been receive, block 2607. Again, it is noted that the acknowledgement message is transmitted over the neighbor's VSP. The CMS receives the acknowledgement message and reassembles the acknowledgement message for transmission to the boot server. The acknowledgement message is then sent to boot server, block 2608. The process of segmentation of blocks by the CMS and reeassembly of the blocks by the booting switch (and visa versa) follows the fragmented TFTP (F-TFTP) protocol. If there are more blocks of boot code to be transmitted, block 2609, the process of the boot server transmitting a block and the information in the block being transmitted using F-TFTP protocol is repeated.

(Chatwani, column 34, lines 30-57)

In column 29, line 28, to column 30, line 2, Chatwani describes that, when a switch (the client) powers up, the switch sends a path access query (PAQ) message. The PAQ message is sent over neighboring switches to the CMS (the boot server). The CMS then responds to the PAQ message from the switch (the client) with a path access response (PAR) message with boot code information. In the alternative embodiment, Chatwani teaches that some other device, other than the CMS, would act as a boot server and provide the boot code to the booting switch. In Figure 26 and the supporting description in column 34, lines 30-57, Chatwani describes a CMS acting as a boot server manager. The CMS does not initiate a request, but, rather, reformats a request from the booting

switch and sends it to the boot server. The boot server responds to the CMS and the CMS reformats the response and sends it to the booting switch. Even if the CMS were considered a client, which it is not, it does not initiate a request for its own configuration, but reformats a request for a booting switch's configuration. In column 23, lines 17-26, Chatwani actually supports Applicants assertion that the CMS is not a client by stating that the CMS registers the client and allows later identification of the client by some address type. Thus, contrary to the Office Action allegation, Chatwani does not teach or suggest that the CMS acts as a client.

Chatwani does not teach or suggest receiving at the client a boot server list if the client configuration information is not found on the first boot server. The Office Action alleges that this feature is taught by the CMS acting as a client in column 29 and the CMS functionality is to determine correct boot code, boot server, optical path and perform load balancing, in column 30, lines 61-67 and column 32, lines1-11, which reads as follows:

> Responsive to receiving the BFQ message, the CMS uses the information provided in the message, including the physical address (field 2141), switch hardware version (field 2142) and switch software version (field 2143) to determine the correct boot code download file for the switch to use. The CMS then responds with a boot file response (BFR) message in the format given by FIG. 21(i) onto the neighbor's VSP boot service channel. The message includes the IP address of a boot server (field 2151) and a boot file name (field 2152). The message is received by the neighbor switch and forwarded on the meta-boot channel to the booting switch in the format given by FIG. 21(j), block 2005.

(Chatwani, column 30, line 61 to column 31, line 5)

> FIG. 21(b) illustrates the format of the PAQ message as it is forwarded on the VPI of a neighbor switch. As can be seen, the VPI field 112, 113 is changed by the neighbor switch's switch fabric to indicate the neighbor switch's VPI and the VCI field 114, 115, 116 is changed, again by the neighbor's switch fabric to indicate the topology service channel (e.g., 2 in the described embodiment) and the port of the neighbor switch on which the PAQ was received. Otherwise, the message, and particularly the information field 102, remains unchanged. Thus, the PAQ message is switched through the neighbor switch without any requirement for intervention by the neighbor switch's controller.
>
> FIG. 21(c) illustrates the format of the PAQ message as it is received by the CMS. During transmission from the neighbor switch to intermediate nodes and onto the CMS, the PAQ's information field is not modified and the only modifications are made in the PAQ's header, by the

> various intermediate switch's switch fabrics, to switch the PAQ along the neighbor's VSP until the PAQ arrives at the master switch where the VPI 112, 113 is set, by the master's switch fabric to a value which uniquely identifies the neighbor of the booting switch.

(Chatwani, column 31, line 64 to column 32, line 18)

In column 30, lines 60-67, Chatwani describes the CMS receiving a boot file query (BFQ) message from a booting switch. The CMS uses the information provided in the BFQ message to determine the correct boot code download file for the switch to use. The CMS then responds to the booting switch with a boot file response (BFR) message. The BFR message includes the IP address of a boot server and a boot file name. Thus, in this section, Chatwani describes the CMS acting as a boot server manager and directs the booting switch to a particular boot server and boot file name. Furthermore, there is no boot server list sent to the booting switch (the client) only one particular boot server and a boot file name. In column 31, line 64 to column 32, line 18, Chatwani describes a path access query (PAQ) message that is sent from the booting switch that is requesting boot code information to the CMS which acts as the boot server. Additionally, this section does not describe a boot server list being returned to the booting switch. Thus, nowhere in these sections, or in any other section, does Chatwani teach or suggest receiving, at the client, a boot server list if the client configuration information is not found on the first boot server.

The Office Action further alleges that Chatwani teaches sending from the client a configuration information request for the client configuration information to each server in the boot server list until the client configuration information is found or a request has been sent to every server in the boot server list at column 26, lines 12-16, column 12, lines 5-6, column 33, lines 16-54, column 32, lines 65-67, which read as follows, and Figures 23a through 24 (not shown).

> Assume that client C1 1402 has provided its logical address as "C1" and client C2 1403 has provided its logical address as "C2". Further assume that client C1 1402 is attached to port 1 of switch Y 1401 and client C2 1402 is attached to port 2 of switch Y 1401. The CMS may then store information providing a one-to-one correspondence between the logical addresses of the clients and their network physical attachment in a client address/location table such as illustrated below:

Logical Address	Switch Module	Port
C1	Y	1
C2	Y	2

(Chatwani, column 26, lines 12-23)

(3) boot services allowing a controller to download software from the supervisor 202 or from a boot file server.

(Chatwani, column 12, lines 5-6)

FIG. 21(h) illustrates the format of the BFQ message as it is received by the CMS. As can be seen, the message is unchanged from the format of FIG. 21(g) except for the VPI field 112, 113 being changed as it is transmitted along the neighbor's VSP, from one intermediate switch to the next intermediate switch and finally to the master switch, where the field is translated by the master's switch fabric to indicate a value which uniquely identifies the neighbor switch. Again, in the described system, this value is simply the switch number of the neighbor switch although in other embodiments it may be a different value in which case there would be a requirement for a mapping table or the like at the CMS to allow the CMS to identify the neighbor switch. It is advantageous to not require the mapping table at least in that more efficient processing can be provided because there is not a need to look up a value in the table.

FIG. 21(i) illustrates the format of a BFR message as it is transmitted by the CMS. As discussed above, the BFR message provides the booting switch with information identifying the boot server and boot file selected by the CMS for booting the booting switch. The boot server is identified in field 2151 and the boot file name is provided in field 2152. Field 2150 provides identification of the output port on which the corresponding BFQ message was transmitted by the booting channel. This field is simply echoed by the CMS from field 2140 of the received BFQ message. The BFR message also provides the booting switch's IP address in field 2154 if the booting switch had set field 2125 to zero. The IP address is assigned by the CMS from a configuration file based on the physical address of the booting switch.

The VPI field 112, 113 is set to indicate the VSP of the neighbor switch and the VCI field 114, 115, 116 is set to indicate the boot service channel number followed by the port number on which the corresponding BFQ was transmitted.

FIG. 21(j) illustrates the format of the BFR message as received by the booting switch. Again, the message is largely unchanged during

> transmission from the CMS except that the VPI field 112, 113 was altered by the switch fabric of the neighbor switch to indicate the meta path number and the VCI field 114, 115, 116 is altered to indicate the meta boot channel number.

(Chatwani, column 33, lines 16-58)

> FIG. 21(f) illustrates the format of a BFQ message as transmitted from the booting switch. As was discussed above, the BFQ message is transmitted on one port of the booting switch, which port is selected by the booting switch. The selection may be based on, among other factors, the cost factor information provided by the PAR messages. As can be seen, the VPI field 112, 113 of the BFQ message as transmitted from the booting switch is set to the meta path number (e.g., 0) and the VCI field 114, 115, 116 is set to the meta boot channel number (e.g., 200). The message body includes 5 fields: (1) identification of the output port on which the BFQ message was sent by the booting switch, field 2140; (2) the IP address of the booting switch, field 2125; (3) the physical address of the booting switch, field 2141; (4) the booting switch hardware version, field 2142; and (5) the booting switch firmware version, field 2143. The CMS will use the information regarding the physical address, hardware version and firmware version to determine the correct boot download file for use by the booting switch. It is noted that if the switch controller does not have its IP address, it will set field 2125 to zero indicating to the CMS that it needs its IP address.

(Chatwani, column 32, line 49 to column 33, line 3)

As discussed above, in contradiction to the Office Action allegation that the CMS is acting as a client, nowhere in any section of the Chatwani reference does the CMS initiate an initial request for client configuration information. Only the booting switch in the Chatwani reference ever initiates a request for its own configuration. In column 26, lines 12-16, Chatwani describes the CMS storing information providing a one-to-one correspondence between the logical addresses of the clients and their network physical attachment in a client address/location table. In column 12, lines 5-6, Chatwani describes boot services allowing a controller to download software from the supervisor or from a boot file server. In column 33, lines 16-54, and column 32, lines 65-67, Chatwani describes the CMS receiving a boot file query (BFQ) message from a booting switch. The CMS uses the information provided in the BFQ message to determine the correct boot code download file for the switch to use. The CMS then responds to the booting switch with a boot file response (BFR) message. The BFR message includes the IP

address of a boot server and a boot file name. Chatwani describes the CMS receiving a boot file query (BFQ) message from a booting switch. Figures 23(a) through 24 do not provide any further teachings of the presently claimed invention. Thus, in these sections, Chatwani describes a booting switch sending a boot file query to the CMS and the CMS directing the booting switch to the server that has the correct boot file. The booting switch of Chatwani does not receive a boot server list and does not sending from the client a configuration information request for the client configuration information to each server in the boot server list until the client configuration information is found or a request has been sent to every server in the boot server list.

The Office Action acknowledges that "Chatwani may not be using the same terms as claimed, such as initiating at a client an initial request for client configuration information; the client information is not found on the first boot server, and sending a boot server list to the client if the information is not found." However the Office Action alleges that "initiating at a client an initial request for client configuration information (it is a part of initialization, in a client-server model client initiates the request and serves the request), if the client information is not found on the first boot server, and sending a boot server list to the client if the information is not found (error handling if very well known in the art, this may be default choice) are very well known in the art." Applicants respectfully traverse this allegation. Applicants respectfully submit that it is not well known in the art to receive at the client a boot server list if the client configuration information is not found on the first boot server; and send from the client a configuration information request for the client configuration information to each server in the boot server list until the client configuration information is found or a request has been sent to every server in the boot server list. This is supported at least in the fact that Chatwani does not perform these features. The Office Action proffets no other evidence in the prior art that these limitations are well-known, or any reasoning to support such a conclusion, other than to summarily dismiss the missing elements as well-known.

The Office Action further alleges that Wiley discloses initiating at a client an initial request for client configuration information and, if the client information is not found on the first boot server, sending a boot server list to the client at column 4, lines 6-24 and lines 50-61, which reads as follows:

> It should be noted that the stifle enable functions to unenable response and conversely a stifle unenable enables the response. Thus, when stifle is enabled, the host does not respond.
>
> The hosts on the server list are then queried directly in order to obtain addresses of the predetermined type of peripheral devices. The client sends a message to the affected hosts to provide lists of addresses of the predetermined type of peripheral device, and compiles a list of these addresses.
>
> The client then sends a second query to the affected hosts which requests lists of addresses of agents. The client then uses the addresses of agents to query the agents to provide the addresses of bootservers for the predetermined type of peripheral device. It is possible for the agent to provide addresses of hosts on the agent's subnet, or to provide all addresses of bootservers for a selected group of peripheral devices of the predetermined type. In the preferred embodiment, the agent provides all addresses of bootservers for the selected group of peripheral devices of the predetermined type.

(Wiley, column 4, lines 3-24)

> The preferred embodiment of the invention utilizes the fact that most network printers and X-terminals utilize one of the following methods to configure an IP Address:
>
> 1. BOOTPD (Boot Protocol Daemon)--Whenever most network peripheral devices power on they broadcast their Link Level Address (LLA) on the network. They do so with the hope that a process on a host on their subnet will be configured to listen for their LLA and respond back to it with the peripheral's IP Address. BOOTPD is one such process. It utilizes the bootptab file to configure its knowledge of LLA/IP Address pairs.

(Wiley, column 4, lines 49-60)

In column 4, lines 3-24, Wiley describes the steps in relation to a client sending out a HALE message to the boot servers connected to a subnet for information pertaining to the existence of data listing address of a predetermined type of peripheral device. In response to the HALE message the boot servers respond with the addresses of the peripheral devices. Once a boot server has responded, the client sends out a stifle message to the servers who have responded so the servers may ignore future HALE messages. After all servers have responded, the client sends out another message to cancel the stifle message. In column 4, lines 51-60, Wiley describes a boot protocol daemon that provides a bootptab file to the peripheral device. However, Wiley further teaches in column 4, lines 39-48, that the response from an agent (computer device) is a

single client-agent relationship, rather than permitting an agent to request addresses from other agents.

Thus, Wiley teaches a client building a list of servers and agents (computer devices) to determine the boot servers that will be used for peripheral devices. Then, as an additional peripheral device is added to the client, the client will automatically know, based on the list, what server or agent to go to directly to find the appropriate boot code. While Wiley may teach initiating at a client an initial request for client configuration information and sending from the client the initial request for client configuration information to a first boot server, that would be the only boot server that the request is sent to as the client has previously built, on its own, a list of the agent or server that has the appropriate boot code.

Wiley does not teach or suggest receiving at the client a boot server list if the client configuration information is not found on the first boot server; and sending from the client a configuration information request for the client configuration information to each server in the boot server list until the client configuration information is found or a request has been sent to every server in the boot server list. As discussed above, the client of Wiley builds the list as a precursor to adding a peripheral device and has no need to send further configuration requests as it knows prior to the peripheral being added on which server the configuration will be stored.

Thus, Chatwani and Wiley, either alone or in combination, are simply not relevant to the claimed invention beyond merely mentioning some of the elements of the presently claimed invention. That is, Chatwani and Wiley, taken alone or in combination, do not teach receiving at the client a boot server list if the client configuration information is not found on the first boot server and sending from the client a configuration information request for the client configuration information to each server in the boot server list until the client configuration information is found or a request has been sent to every server in the boot server list. Thus, Applicants respectfully submit that Chatwani does not teach all of the features of independent claims 1, 14 and 25.

Independent claims 10, 21 and 26 recite similar features in their respective claim terminology. For example, claim 10, which is representative of the other rejected independent claims 21 and 26 with regard so similarly recited subject matter, recites

"receiving at a boot server an initial request for client configuration information from a client, wherein the initial request is initiated at a client and sending from the boot server the boot server list to the client if the client configuration information is not found."

Thus, Chatwani and Wiley, taken alone or in combination, do not teach or suggest all of the features in independent claims 1, 10, 14, 21, 25 and 26 as is required under 35 U.S.C. § 103(a). At least by virtue of their dependency on independent claims 1, 10, 14 and 21, the specific features of claims 2-9, 11-13, 15-20 and 22-24 are not taught by Chatwani and Wiley, either alone or in combination. Accordingly, Applicants respectfully request withdrawal of the rejection of claims 1-26 under 35 U.S.C. § 103(a).

Furthermore, there is not so much as a suggestion in either reference to modify the references to include such features. That is, there is no teaching or suggestion in Chatwani or Wiley that a problem exists for which receiving at the client a boot server list if the client configuration information is not found on the first boot server and sending from the client a configuration information request for the client configuration information to each server in the boot server list until the client configuration information is found or a request has been sent to every server in the boot server list, is a solution.

Moreover, neither reference teaches or suggests the desirability of incorporating the subject matter of the other reference. That is, there is no motivation offered in either reference for the alleged combination. The Office Action alleges that the motivation for the combination is "to have system where client can obtain an alternate boot sever by requesting the boot server list in the case of primary boot server is affected." As discussed above, Chatwani merely uses a CMS to operate as a boot server or a boot server manager and maintains a boot server list. Wiley merely describes building a list of boot servers on the client and then referencing the list as additional peripheral devices are added. Neither reference receives at the client a boot server list if the client configuration information is not found on the first boot server and sends from the client a configuration information request for the client configuration information to each server in the boot server list until the client configuration information is found or a request has been sent to every server in the boot server list. Thus, the only teaching or suggestion to even attempt the alleged combination is based on a prior knowledge of Applicants' claimed invention

thereby constituting impermissible hindsight reconstruction using Applicants' own disclosure as a guide.

II. Conclusion

It is respectfully urged that the subject application is patentable over the prior art of record and is now in condition for allowance. The Examiner is invited to call the undersigned at the below-listed telephone number if in the opinion of the Examiner such a telephone conference would expedite or aid the prosecution and examination of this application.

Respectfully submitted,

DATE: April 4, 2005

Francis Lammes
Reg. No. 55,353
Yee & Associates, P.C.
P.O. Box 802333
Dallas, TX 75380
(972) 385-8777
Agent for Applicants